

Optimizing earthquake damage prediction using particle swarm optimization-based feature selection

Nurul Anisa Sri Winarsih¹, Ricardus Anggi Pramunendar¹, Guruh Fajar Shidik¹, Budi Widjajanto², Muhammad Syaifur Rohman¹, Danny Oka Ratmana¹

¹Department of Informatics Engineering, Faculty of Computer Science, Universitas Dian Nuswantoro, Semarang, Indonesia

²Department of Information System, Faculty of Computer Science, Universitas Dian Nuswantoro, Semarang, Indonesia

Article Info

Article history:

Received Feb 29, 2024

Revised Aug 29, 2024

Accepted Sep 28, 2024

Keywords:

Earthquake damage prediction
Feature selection
Machine learning
Particle swarm optimization
Phasor particle swarm optimization

ABSTRACT

Earthquakes have destroyed the economy and killed many people in many countries. Emergency response actions immediately after an earthquake significantly reduce economic losses and save lives, so accurate earthquake damage predictions are needed. This research looks at how machine learning (ML) techniques are used to predict damage from earthquakes. The ML algorithms used are k-nearest neighbors (KNN), decision tree (DT), random forest (RF), and Naïve Bayes (NB). Feature selection is necessary, it needs to select the most relevant features from big data. One of the most commonly used algorithms to optimize ML is particle swarm optimization (PSO). PSO is also suitable for feature selection. This research compares various of PSO. Based on research, the RF algorithm with Phasor PSO has the highest fitness score. This process succeeded in reducing features from 38 features to 14 features. Based on the process after feature selection, it was found that the KNN, DT, and RF algorithms had improved. RF obtained the best accuracy, namely 72.989%. The processing time in DT, RF, and NB is faster than before. In conclusion, the ML algorithm can be combined with PSO feature selection to create a classification model that provides better performance than without feature selection.

This is an open access article under the [CC BY-SA](#) license.



Corresponding Author:

Ricardus Anggi Pramunendar

Department of Informatics Engineering, Faculty of Computer Science, Universitas Dian Nuswantoro
Semarang, 50131, Jawa Tengah, Indonesia

Email: ricardus.anggi@dsn.dinus.ac.id

1. INTRODUCTION

Earthquakes have destroyed the economy of citizens and killed many people in many countries [1]. An example of the largest earthquake with a magnitude of 9.5 on the Richter scale occurred in Valdivia, Chile, on May 22, 1960. The victims of this earthquake claimed 1,655 deaths, 3,000 were injured, and two million people were displaced. This disaster caused death and destruction in Hawaii, Japan, and the Philippines and caused a tsunami. The resulting losses amounted to US\$550 million in Chile [2]. Emergency response actions immediately after an earthquake significantly reduce economic losses and save lives, so accurate and real-time estimates of seismic damage are needed [3]. Delayed or inaccurate seismic damage estimates can have severe consequences [4].

Developing techniques to predict disaster damage can reduce casualties and economic losses [5]. Various research and studies have been carried out that focus on predicting multiple types of natural disasters, such as earthquakes [1], [6], [7] landslides [5], [8] liquefaction [9], rock explosions [10], drought, floods, and storms [11]. This research focuses on exploring the application of various machine learning (ML) methods in predicting earthquake damage. Various ML algorithms, such as k-nearest neighbors (KNN), random forest

(RF), decision tree (DT), and Naïve Bayes (NB), are often used in multiple fields, including research discussing disasters. For example, KNN is used in earthquake prediction models [12], the RF algorithm in disaster prediction [13], as well as DT and NB, which are also applied in similar disaster prediction analyses [14].

In the context of ML, feature selection is a necessary process. Need to select the most relevant features from big data. Relevant features can reduce computational complexity and improve temporal models' accuracy. One of the most commonly used algorithms to optimize ML is particle swarm optimization (PSO). PSO suits multiobjective optimization challenges by efficiently exploring complex and large-dimensional solution spaces. The PSO process searches for ideal solutions by mimicking the behavior of a flock of birds or a swarm of particles when searching for a perfect solution, repeatedly changing candidate solutions based on their previous performance and the performance of their neighbors in the solution space. PSO can be used as a powerful and flexible multiobjective optimization technique [15]. Implementing PSO can increase the accuracy of image datasets [16] and be used for random noise reduction of seismic activity in deserts [17].

Research about predicted seismic damage on a regional scale using a data set of building damage during the 2015 Nepal Gorkha earthquake was done using the ML method, namely random forest regression (RFR). This research classifies damage into three classes, green, amber, and red, based on the level of post-earthquake damage. The results obtained an average accuracy of 0.68, which shows that limiting learning to basic building characteristics such as number of floors, height, base area, and building age produces reliable predictions of building damage [18]. The feature selection technique can increase the accuracy of predicted earthquakes. This research applied various algorithms or feature selection techniques based on different searches, such as population, local, ranking, and evaluator-based searches, such as correlation, consistency, and distance matrices on earthquake data for 47 years in Southern California. The result is that the number of existing attributes can be reduced by up to 80%, thereby improving the matrix from the original [1]. This research will combine the power of ML in classification with the efficiency of PSO as feature selection, allowing the identification of the most relevant and influential features in predicting earthquake damage. By focusing analysis on critical features, this research improves the accuracy and efficiency of predictive models while providing deep insight into crucial factors influencing seismic damage.

2. PROPOSED METHOD

The research begins by preparing a dataset from earthquake damage data. The dataset is separated into training and testing data, then processed using ML with the original data. Next, the researchers selected features using original PSO, adaptive inertia weight (AIW) PSO, linearly decreasing weight-particle swarm optimization (LDW PSO), and phasor PSO. After getting optimal features, training, and testing data are processed using ML again. At the end of the process is a diagnosis result, where researchers can determine whether ML and PSO can increase accuracy. The proposed model diagram is illustrated in Figure 1.

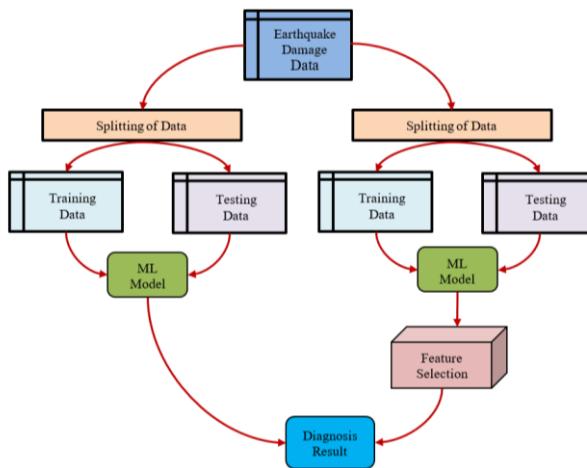


Figure 1. Diagram of proposed model

2.1. Dataset

The data that researchers use is global data that Subhash Ghimire has processed. This data was formed based on the Gorkha earthquake in Gandaki Pradesh, Nepal, at April 2015, with a magnitude of 7.8

on the Richter scale. The resulting losses amounted to \$10 billion, about half of Nepal's nominal GDP. As many as 9,000 people died, millions of people were left homeless [18].

The data was collected by the Central Bureau of Statistics and Kathmandu Living Labs under the National Planning Commission Secretariat of Nepal. It is one of the most enormous post-disaster datasets ever assembled. This data offers insightful data on home circumstances, the effects of the earthquake, and socioeconomic and demographic statistics. This dataset has 40 columns or features and 260,601 rows of data. The building_id feature is not used because it contains a unique identifier. The damage_grade feature is used as a label. So, 38 features need to be processed. Other features consist of socio-economic-demographic statistics, earthquake impacts, and household conditions. The dataset features are described in Table 1.

Table 1. Dataset's features

No	Feature and description
1	geo_level_1_id Building location is indicated by the geo level 1: 0 – 30
2	geo_level_2_id Building location is indicated by the geo level 2: 0 – 1427
3	geo_level_3_id Building location is indicated by the geo level 3: 0 – 12567
4	count_floors_pre_eq the number of floors the structure had prior to the seismic activity
5	age the building's age expressed in years
6	area_percentage represents the building footprint's normalized area
7	height_percentage the building footprint's normalized height.
8	land_surface_condition the state of the land on which the structure was constructed. n, o, and t are possible values
9	foundation_type the kind of foundation that is employed in construction. h, i, r, u, and w are possible values
10	roof_type the kind of roof that was built. It is possible for n, q, and x
11	ground_floor_type the ground floor's kind
12	other_floor_type the kind of buildings used on levels higher than the ground (apart from the roof). Possible values: f, m, v, x, and z.
13	position the building's position are possible values: j, o, s, t.
14	plan_configuration building plan configuration; possible values: a, c, d, f, m, n, o, q, s, u are examples of possible values for the flag variables
15	has_superstructure_adobe_mud
16	has_superstructure_mud_mortar_stone
17	has_superstructure_stone_flag
18	has_superstructure_cement_mortar_stone
19	has_superstructure_mud_mortar_brick
20	has_superstructure_cement_mortar_brick
21	has_superstructure_timber
22	has_superstructure_bamboo
23	has_superstructure_rc_non_engineered
24	has_superstructure_rc_engineered
25	has_superstructure_other binary variable specifies construct the superstructure
26	legal_ownership_status the land's legal ownership status at the time a building was constructed
27	count_families the number of families residing in the building; possible values are a, r, v, and w
28	has_secondary_use
29	has_secondary_use_agriculture
30	has_secondary_use_hotel
31	has_secondary_use_rental
32	has_secondary_use_institution
33	has_secondary_use_school
34	has_secondary_use_industry
35	has_secondary_use_health_post
36	has_secondary_use_gov_office
37	has_secondary_use_use_police
38	has_secondary_use_other binary variable specifies whether the building was utilized for any secondary purposes damage_grade (LABEL)
	A prediction label for earthquake damage prediction. There are three levels of damage: i) indicates minimal harm, ii) indicates a moderate degree of harm, and iii) indicates nearly total devastation

2.2. Machine learning algorithm

2.2.1. K-nearest neighbors

KNN is a simple and widely used ML algorithm for classification and regression. This algorithm is included in the ML category, which does not require training (non-parametric) and is instance-based [19]. This algorithm identifies the closest 'k' data points (neighbors) of an unknown data point. This approach categorizes new data points according to the majority class of their 'k' nearest neighbors. The most common distance formula is Euclidean distance, which is calculated as (1):

$$\text{Euclidean distance} = \sqrt{\sum_{i=1}^n (x_i - y_i)^2} \quad (1)$$

where x_i and y_i are the coordinates of two points in n-dimensional space. This algorithm selects 'k' nearest neighbors based on this distance, and the class or output value of the new point is determined based on these data. Other distances that can be used include Manhattan, Minkowski, and others, depending on the use case.

2.2.2. Random forest

RF is a ML algorithm for regression, classification, and other predictive modeling tasks [20]. This algorithm is a type of ensemble learning where the algorithm combines the results of various individual models to improve overall performance. How RF work:

- Making DT: creating several DT using 'bagging' (bootstrap aggregating), where each tree is built from a random sample with a replacement from the original dataset.
- Random feature selection: at each split in the DT, only a limited number of features are randomly selected.
- Prediction combining: predictions from each DT are combined to create a final prediction. Classification is usually done through majority voting, whereas the average of the projections is used in regression.

This algorithm relies more on creating and merging DT than specific mathematical calculations. RF effectiveness lies in its ability to reduce overfitting while maintaining high accuracy, making it a popular choice for various machine-learning applications.

2.2.3. Decision tree

DT are methods rooted in tree-like structures used in data mining and ML for decision-making processes. This method operates by taking a set of input attributes and producing a Boolean decision as its output. Each route in the tree represents a series of data splits, culminating in a Boolean result found at the final node, known as a leaf node. Its application in practical scenarios is well known due to its ease of interpretation in both human and programming languages [21]. DT are often used in various applications, from classification and regression to detecting important features in data sets. DT are built through an algorithmic process that breaks down data based on certain criteria. However, there are key concepts and calculations involved in making a DT, such as:

- Node splitting: decisions about how to split nodes are based on metrics such as Gini Impurity and entropy as (2) and (3):

$$\text{Gini Impurity} = 1 - \sum_{i=1}^c (p_i)^2 \quad (2)$$

$$\text{Entropy} = - \sum_{i=1}^c p_i \cdot \log_2(p_i) \quad (3)$$

where: c is number of classes in the dataset and p_i is proportion of samples belonging to class i in the dataset.

- Tree pruning: this is not a formula but rather a process for removing parts of the tree that may be based on noise or overfitting.

These calculations help determine where divisions should be made in the tree and how deep the tree should grow. The actual structure of a DT is built algorithmically by applying these calculations repeatedly to break down the data in the most effective way until a stopping criterion is met.

2.2.4. Naïve Bayes

Naive Bayes is a classification method in ML that is based on Bayes' Theorem. This method is nicknamed "naive" because of its simple assumption, namely that the features in a dataset are considered independent of each other. It means that other features do not influence the presence or absence of a feature in the same class. This approach has proven to be very efficient, especially in processing large datasets and is often used for text classification tasks, including spam filtering and sentiment analysis. The Naive Bayes algorithm works by utilizing training data to calculate probabilities. Then, use the probability of each attribute from the test data for all existing classes. The class with the highest probability is then selected as the label of the test data in question, with the Bayes Theorem as (4):

$$P(X|H) = \frac{P(X|H) \cdot P(H)}{P(X)} \quad (4)$$

where $P(X|H)$ is the conditional probability of hypothesis H, considering that data X has occurred. This value is called the posterior probability, which is the probability of a hypothesis after considering the evidence. Meanwhile, $P(X|H)$ is the probability that data X will occur if the hypothesis H is true, known as the likelihood. $P(H)$ is the initial probability of the hypothesis before looking at the data or prior probability. Finally, $P(X)$ is the total probability of data X and is known as marginal probability or evidence [12].

2.2.5. Evaluation

The classification evaluation process produces a confusion matrix. The matrix consists of the number of correct predictions, namely true positive (TP) and true negative (TN), as well as the number of incorrect predictions, namely false positive (FP) and false negative (FN). TP consists of the number of positive detected positive data. TN refers to the number of actual negative data. FP is the number of negative data that were incorrectly classified as positive. In contrast, FN is the number of positive incidents categorized as negative. Accuracy provides an idea of how much a classification model can predict correctly. Accuracy is calculated from the number of TP and TN divided by the total number of predictions [12] as (5):

$$\text{Accuracy} = \frac{TP+TN}{TP+TN+FP+FN} \quad (5)$$

2.3. Particle swarm optimization

As explained in the study, Eberhart and Kennedy describe PSO as a stochastic search method that uses a group of individuals to simulate the behaviour of a flock of birds to achieve an optimal solution. PSO is an approach in relatively modern swarm intelligence where each particle in the swarm iteratively adjusts its trajectory based on its personal best and overall swarm experience to navigate towards a globally optimal solution. Each particle in the PSO iteration updates its position and velocity based on the extreme values that have been recorded [22].

The optimization procedure in PSO goes through various stages, such as initializing each particle randomly, evaluating the fitness of each particle based on a predetermined fitness function [23], updating the speed and position of the particle based on its best position (Pbest) and the best position of the group (Gbest) and repeats the initialization and fitness evaluation process until the termination condition is met. Where the fitness function in PSO is a measure of how well a particle solution performs in solving an optimization problem by evaluating the quality of the particle's position in the search space. The fitness function is usually defined based on the specific problem being solved, which can be a mathematical function that calculates the goal or objectives of the optimization problem. In this way, the stages above will be carried out in research on the use of PSO, as shown in Figure 2.

2.3.1. Adaptive inertia weight particle swarm optimization

As has been explained, performance on PSO is greatly influenced by the inertial weight parameter. In traditional PSO, the inertia weight is often set as a constant, so it is considered inefficient. Therefore, AIW is used as a mechanism to regulate the influence of particle speed from the previous iteration on the speed in the current iteration. Using AIW means the values in the inertia weights can adjust dynamically throughout the search process. The goal is to enable particles to explore the search space more efficiently by maintaining global exploration early in the search process with higher inertial weights and then increasing local exploitation capabilities by reducing inertial weights in response to information obtained during the search [24].

2.3.2. Linearly decreasing inertia weight particle swarm optimization

LDW PSO is a variant of the PSO algorithm that changes the weight inertia value linearly during iteration. At the beginning of the iteration, the LDW PSO weight inertia value increases to encourage exploration of a wider search space. However, as the weight inertia linearly decreases, the particles can concentrate more to exploit promising areas. LDW PSO has attracted attention in several contexts, one of which is feature selection, where its ability to adapt to changes in convergence speed can help find more suitable features [25].

2.3.3. Adaptive inertia weight particle swarm optimization

PSO is a creative version of the PSO algorithm that introduces the idea of phases or Phasors in the movement of particles. This phase shows angles in complex coordinates and replaces conventional velocity and position values. This principle is used by PPSO to provide more adaptive directions and velocities to particles in the search space. In this way, P PSO can explore the search space efficiently by dynamically adjusting the particle velocity based on the surrounding information. In addition, the problem of slow convergence can be easily overcome by P PSO. PSO P's ability to adapt to high levels of complexity and dimensionality [26].

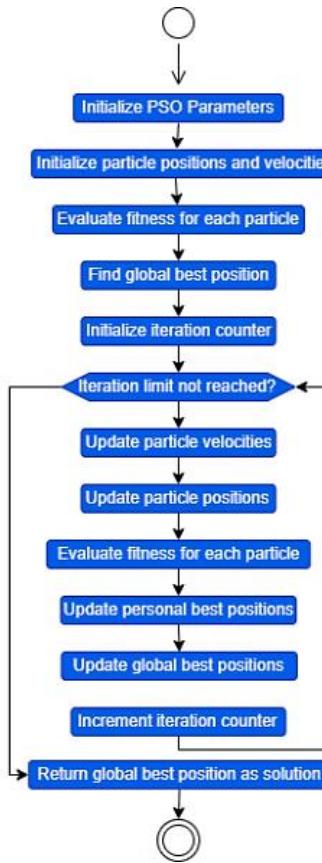


Figure 2. PSO model flowchart

3. RESULTS AND DISCUSSION

In this chapter, we will present a comparison of the results of feature selection using PSO. Then a comparison of the best accuracy will be carried out between ML algorithms without feature selection and with feature selection. There are ten epochs used, including 5, 10, 15, 20, 25, 30, 35, 40, 45, and 50. The comparison results between the ML algorithm and various PSO selection features are in Tables 2 to 5.

Table 2. Fitness score using original PSO

ML algorithm	Fitness score per epoch/time (minutes)									
	5	10	15	20	25	30	35	40	45	50
KNN	0.712	0.713	0.710	0.709	0.711	0.713	0.711	0.714	0.714	0.713
	108.143	200.236	269.260	320.220	371.179	396.587	441.237	521.224	536.347	573.270
DT	0.714	0.708	0.713	0.711	0.716	0.711	0.712	0.711	0.721	0.715
	0.165	0.450	1.230	1.331	1.617	1.685	1.901	1.940	1.994	2.150
RF	0.713	0.711	0.708	0.717	0.718	0.727	0.721	0.722	0.725	0.718
	5.052	8.277	12.313	16.121	19.158	23.516	28.384	31.011	35.753	37.717
NB	0.429	0.574	0.575	0.572	0.572	0.574	0.573	0.573	0.575	0.575
	0.132	0.309	0.434	0.584	0.632	0.759	0.887	0.944	0.998	1.040

Table 3. Fitness score using AIW PSO

ML algorithm	Fitness score per epoch/time (minutes)									
	5	10	15	20	25	30	35	40	45	50
KNN	0.708	0.711	0.711	0.711	0.712	0.711	0.712	0.712	0.711	0.715
	91.276	184.351	230.059	287.779	322.702	394.364	427.094	484.741	481.910	560.380
DT	0.677	0.703	0.710	0.712	0.720	0.713	0.714	0.716	0.714	0.715
	0.179	0.300	0.456	0.551	0.694	0.860	0.896	1.251	1.264	1.350
RF	0.709	0.721	0.720	0.711	0.716	0.720	0.717	0.720	0.715	0.715
	4.368	7.739	12.515	16.556	17.930	22.470	28.190	31.891	38.258	36.893
NB	0.574	0.576	0.573	0.569	0.575	0.575	0.574	0.574	0.574	0.575
	0.175	0.278	0.450	0.540	0.638	0.750	0.800	0.908	0.958	0.998

Table 4. Fitness score using LDW PSO

ML algorithm	Fitness score per epoch/time (minutes)									
	5	10	15	20	25	30	35	40	45	50
KNN	0.706	0.710	0.712	0.716	0.713	0.712	0.712	0.714	0.711	0.713
	96.485	150.469	232.339	279.876	306.199	399.952	420.486	481.948	500.390	525.624
DT	0.699	0.711	0.711	0.715	0.725	0.723	0.719	0.714	0.720	0.715
	0.640	0.733	0.889	1.065	1.295	1.328	1.344	1.633	1.768	1.764
RF	0.707	0.716	0.713	0.719	0.720	0.716	0.713	0.722	0.724	0.718
	5.052	8.277	12.313	16.121	19.158	23.516	28.384	31.011	35.753	37.717
NB	0.549	0.574	0.574	0.575	0.571	0.575	0.572	0.575	0.574	0.575
	0.176	0.298	0.427	0.582	0.609	0.756	0.791	0.884	1.012	1.022

Table 5. Fitness score using Phasor PSO

ML algorithm	Fitness score per epoch/time (minutes)									
	5	10	15	20	25	30	35	40	45	50
KNN	0.709	0.712	0.710	0.710	0.711	0.711	0.712	0.713	0.712	0.714
	44.981	103.361	214.953	292.668	154.886	484.893	531.359	596.694	374.391	520.256
DT	0.715	0.672	0.696	0.712	0.715	0.707	0.719	0.712	0.714	0.722
	0.179	0.280	0.436	0.641	0.741	0.801	1.085	1.050	1.191	1.326
RF	0.711	0.714	0.710	0.720	0.716	0.718	0.722	0.727	0.720	0.723
	4.004	8.777	12.912	17.081	18.072	25.035	28.116	29.232	35.306	38.996
NB	0.558	0.570	0.572	0.575	0.572	0.573	0.567	0.574	0.574	0.574
	0.165	0.288	0.421	0.562	0.562	0.562	0.802	0.802	0.963	0.990

In Table 2, we can see a comparison of the KNN, DT, and RF ML algorithms against feature selection using the original PSO. The highest fitness score for the KNN algorithm is 0.714, while the highest fitness score for the DT algorithm is 0.721. The two highest fitness scores were at epoch 45. Then, the RF algorithm obtained the highest fitness score of 0.727 at epoch 30. The score on the DT was the highest value. The highest fitness score in NB is 0.575. This value is the smallest fitness score compared to other algorithms. Even so, NB can complete the feature selection task very quickly, followed by DT, RF, and the longest KNN.

The results from AIW PSO can be seen in Table 3. The highest fitness score obtained by RF was 0.721 at epoch 10. Then, the DT algorithm was 0.720 at epoch 25 and 0.715 for the KNN algorithm. NB obtained the smallest fitness score with a value of 0.576 at the 10th epoch. Meanwhile, the fastest time was obtained by Naive Bayes with a maximum process of 0.998 minutes, followed by DT 1,350 minutes, RF 36,893 minutes, and KNN 560.38 minutes.

In contrast to the previous table, the highest LDW PSO is obtained by DT with a value of 0.725 at epoch 25. Then, the RF algorithm with a fitness score of 0.724 at epoch 45. Followed by KNN with a fitness value of 0.716 at epoch 20. Lastly is NB, which has a value of 0.575 at epoch 50. The speed time is still the same as the previous sequence, namely NB, DT, RF, and KNN.

The highest Phasor PSO is found in the RF algorithm at epoch 40 with a value of 0.727 at epoch 40. This fitness score value is the largest compared to other PSO algorithms. Then followed by DT 0.722 epoch 50, then KNN 0.714 epoch 50, and finally NB 0.575 epoch 20. The fastest time is NB, then DT, RF, and the longest is KNN. Of the 4 PSO algorithms that have been run, Phasor PSO is the fastest algorithm with a processing time of 0.990 minutes, while AIW PSO is the shortest at 0.998 minutes, LDW PSO 1.022 minutes and original PSO 1.040. The NB algorithm obtained all the fastest times.

A summary of Tables 2 to 5 is in Table 6. The highest fitness score is 0.7272 in the RF algorithm with Phasor PSO and produces 14 feature selections. Then, the DT algorithm with LDW PSO with a value of 0.725 produces 20 selected features. Followed by KNN with LDW PSO value 0.716, finding 24 feature selections. Finally, NB with AIW PSO of 0.576 produces 19 feature selections. Based on the highest fitness score, researchers used 14 feature selections. These features include: geo_level_1_id, geo_level_2_id, geo_level_3_id, foundation_type, has_superstructure_adobe_mud, has_superstructure_stone_flag, has_superstructure_mud_mortar_brick, has_superstructure_cement_mortar_brick, has_superstructure_timber, legal_ownership_status, has_secondary_use_agriculture, has_secondary_use_health_post, has_secondary_use_gov_office, and has_secondary_use_use_police. Then the researcher will compare before and after using feature selection. Table 7 shows the accuracy of the ML algorithm with 38 complete features and compared with the accuracy after feature selection with 14 features. The KNN, DT, and RF algorithms have experienced improvements. DT before feature selection gets an accuracy of 65.722%, increasing to 72.753%, with a difference of 7.031%. Then the KNN algorithm rose from 67.282% to 2.958% become 70.240%. RF obtained the highest accuracy value from 71.729% to 72.989%, an increase of 1.260%. The smallest accuracy was NB, from 57.287%, decreasing to 57.017%. NB decreased -0.270%.

Table 6. Best fitness score using All PSO

ML algorithm	Original PSO	Best fitness score			Number of features after feature selection
		AIW PSO	LDW PSO	Phasor PSO	
KNN	0.714	0.715	0.716	0.714	24
DT	0.721	0.720	0.725	0.722	20
RF	0.727	0.721	0.7247	0.727	14
NB	0.575	0.576	0.575	0.575	19

Table 7. Result after and before

ML algorithm	Before feature selection (%)	Time (minutes)	After feature selection (%)	Time (minutes)	Difference accuracy
KNN	67.282	0.127	70.240	1.253	2.958
DT	65.722	14.434	72.753	6.345	7.031
RF	71.729	0.586	72.989	0.310	1.260
NB	57.287	0.004	57.017	0.001	-0.270

The processing time before and after feature selection in DT, RF, and NB is accelerated. The fastest algorithm is NB from 0.004 to 0.0001 minutes. The second fastest was obtained by RF from 0.586 to 0.310 minutes. The longest algorithm obtained by the DT was from 14,434 to 6,345 minutes. In contrast to the KNN algorithm, after feature selection, it takes longer, from 0.127 to 1.253 minutes.

4. CONCLUSION

In this research, a comparison of the accuracy results of earthquake damage prediction was carried out with the Gorkha earthquake dataset without feature selection and after feature selection. The feature selection algorithm used is PSO, by comparing original PSO, AIW PSO, LDW PSO, and Phasor PSO. Feature selection using PSO is processed using epochs 5, 10, 15, 20, 25, 30, 35, 40, 45, and 50. The classification algorithms used are KNN, DT, RF, and NB. Based on research, the RF algorithm with Phasor PSO has the highest fitness score of 0.7272 at the 40th epoch. From this process, the Phasor PSO algorithm succeeded in reducing features from 38 features to 14 features.

Based on the process after feature selection, it was found that the KNN, DT, and RF algorithms had improved. The biggest increase was obtained by DT from 65.722% to 72.753%, with a difference of 7.031%. Meanwhile, the best accuracy was obtained by RF from 71.729% to 72.989%. In the NB algorithm, it decreased from 57.287% to 57.017%. NB decreased -0.270%. It happens because the features that are removed during feature selection turn out to be important for Naive Bayes classification, causing a decrease in accuracy. The processing time before and after feature selection in DT, RF, and NB is accelerated. In contrast to the KNN algorithm, after feature selection, it takes longer, from 0.127 to 1.253 minutes. KNN relies heavily on the distance between observations in the feature space. If certain features removed during feature selection affect inter-observation distances, distance calculations can become more difficult and require more processing time

In the future, this research work may include several areas that can be improved and expanded. Some recommendations for future work are as follows: using deep learning algorithms to enhance model performance and learning various hyperparameter tuning. It further improves accuracy by combining pre-trained models, such as BERT or GPT. As well as building a real-time earthquake damage prediction system so that it can immediately provide emergency response measures after an earthquake occurs.

ACKNOWLEDGEMENTS

This research supported by Intelligent Distributed Surveillance System (IDSS) of Universitas Dian Nuswantoro and funded by *Lembaga Penelitian dan Pengabdian Kepada Masyarakat* (LPPM) of Universitas Dian Nuswantoro with Grant ID 109/A.38-04/UDN-09/XI/2023.

REFERENCES

- [1] K. M. Asim, A. Idris, T. Iqbal, and F. Martínez-Álvarez, "Earthquake prediction model using support vector regressor and hybrid neural networks," *PLoS ONE*, vol. 13, no. 7, Jul. 2018, doi: 10.1371/journal.pone.0199004.
- [2] C. Pastén *et al.*, "The role of site conditions on the structural damage in the city of Valdivia during the 22 May 1960Mw 9.5 Megathrust Chile Earthquake," *Seismological Research Letters*, vol. 92, no. 6, pp. 3437–3451, Nov. 2021, doi: 10.1785/0220190321.
- [3] X. Yuan, D. Tanksley, L. Li, H. Zhang, G. Chen, and D. Wunsch, "Faster post-earthquake damage assessment based on 1D convolutional neural networks," *Applied Sciences (Switzerland)*, vol. 11, no. 21, pp. 1–14, Oct. 2021, doi: 10.3390/app11219844.
- [4] Y. Xu, X. Lu, Y. Tian, and Y. Huang, "Real-time seismic damage prediction and comparison of various ground motion intensity

measures based on machine learning," *Journal of Earthquake Engineering*, vol. 26, no. 8, pp. 4259–4279, Jun. 2022, doi: 10.1080/13632469.2020.1826371.

[5] L. Zhang, B. Shi, H. Zhu, X. B. Yu, H. Han, and X. Fan, "PSO-SVM-based deep displacement prediction of Majiagou landslide considering the deformation hysteresis effect," *Landslides*, vol. 18, no. 1, pp. 179–193, Jan. 2021, doi: 10.1007/s10346-020-01426-2.

[6] J. Roiz-Pagador, A. Chacon-Maldonado, R. Ruiz, and G. Asencio-Cortes, "Earthquake prediction in California using feature selection techniques," in *16th International Conference on Soft Computing Models in Industrial and Environmental Applications (SOCO 2021)*, 2022, pp. 728–738. doi: 10.1007/978-3-030-87869-6_69.

[7] G. F. Shidik, *et al.*, "LUTanh activation function to optimize BI-LSTM in earthquake forecasting," *International Journal of Intelligent Engineering and Systems*, vol. 17, no. 1, pp. 572–583, Feb. 2024, doi: 10.22266/ijies2024.0229.48.

[8] X. Yu, Y. Wang, R. Niu, and Y. Hu, "A combination of geographically weighted regression, particle swarm optimization and support vector machine for landslide susceptibility mapping: a case study at Wan Zhou in the three gorges area, China," *International Journal of Environmental Research and Public Health*, vol. 13, no. 5, pp. 1–35, May 2016, doi: 10.3390/ijerph13050487.

[9] J. Jin, S. Yuan, H. Cui, X. Xiao, and B. Jia, "A threshold model of tailings sand liquefaction based on PSO-SVM," *Sustainability (Switzerland)*, vol. 14, no. 5, pp. 1–17, Feb. 2022, doi: 10.3390/su14052720.

[10] J. Zhou, P. Yang, P. Peng, M. Khandelwal, and Y. Qiu, "Performance evaluation of rockburst prediction based on PSO-SVM, HHO-SVM, and MFO-SVM hybrid models," *Mining, Metallurgy and Exploration*, vol. 40, no. 2, pp. 617–635, Feb. 2023, doi: 10.1007/s42461-022-00713-x.

[11] V. Panwar and S. Sen, "Disaster damage records of EM-DAT and DesInventar: a systematic comparison," *Economics of Disasters and Climate Change*, vol. 4, no. 2, pp. 295–317, Jul. 2020, doi: 10.1007/s41885-019-00052-0.

[12] S. Mangalathu, H. Sun, C. C. Nweke, Z. Yi, and H. V. Burton, "Classifying earthquake damage to buildings using machine learning," *Earthquake Spectra*, vol. 36, no. 1, pp. 183–208, Feb. 2020, doi: 10.1177/8755293019878137.

[13] B. Xiao, *et al.*, "Combined SBAS-InSAR and PSO-RF algorithm for evaluating the susceptibility prediction of landslide in complex mountainous area: a case study of Ludian County, China," *Sensors*, vol. 22, no. 20, pp. 1–24, Oct. 2022, doi: 10.3390/s2208041.

[14] S. Mangalathu, S. H. Hwang, E. Choi, and J. S. Jeon, "Rapid seismic damage evaluation of bridge portfolios using machine learning techniques," *Engineering Structures*, vol. 201, Dec. 2019, doi: 10.1016/j.engstruct.2019.109785.

[15] A. Gudur, V. Patil, L. Gopal, and S. Thapliyal, "Multi-objective optimization for breast cancer risk prediction models with particle swarm optimization," *International Journal of Intelligent Systems and Applications in Engineering*, vol. 12, no. 3s, pp. 531–541, 2024.

[16] E. A. Pambudi, P. N. Andono, and R. A. Pramunendar, "Image segmentation analysis based on k-means PSO by using three distance measures," *ICTACT Journal on Image and Video Processing*, vol. 9, no. 1, pp. 1821–1826, Aug. 2018, doi: 10.21917/ijivp.2018.0256.

[17] M. Li, Y. Li, N. Wu, Y. Tian, and T. Wang, "Desert seismic random noise reduction framework based on improved PSO-SVM," *Acta Geodaetica et Geophysica*, vol. 55, no. 1, pp. 101–117, Mar. 2020, doi: 10.1007/s40328-019-00283-3.

[18] S. Ghimire, P. Guéguen, S. Giffard-Roisin, and D. Schorlemmer, "Testing machine learning models for seismic damage prediction at a regional scale using building-damage dataset compiled after the 2015 Gorkha Nepal earthquake," *Earthquake Spectra*, vol. 38, no. 4, pp. 2970–2993, Nov. 2022, doi: 10.1177/87552930221106495.

[19] A. Novianty, C. Machbub, S. Widiyantoro, I. Meilano, and H. Irawan, "Tsunami potential identification based on seismic features using KNN algorithm," in *Proceeding - 2019 IEEE 7th Conference on Systems, Process and Control, ICSPC 2019*, 2019, pp. 155–160. doi: 10.1109/ICSPC47137.2019.9068095.

[20] H. Zhang, J. Zhou, D. J. Armaghani, M. M. Tahir, B. T. Pham, and V. Van Huynh, "A combination of feature selection and random forest techniques to solve a problem related to blast-induced ground vibration," *Applied Sciences (Switzerland)*, vol. 10, no. 3, pp. 1–18, Jan. 2020, doi: 10.3390/app10030869.

[21] F. J. Yang, "An extended idea about decision trees," in *Proceedings - 6th Annual Conference on Computational Science and Computational Intelligence, CSCI 2019*, IEEE, Dec. 2019, pp. 349–354. doi: 10.1109/CSCI49370.2019.00068.

[22] Rajani, D. Kumar, and V. Kumar, "Impact of controlling parameters on the performance of MOPSO algorithm," *Procedia Computer Science*, vol. 167, pp. 2132–2139, 2020, doi: 10.1016/j.procs.2020.03.261.

[23] R. Kalimuthu and B. Thomas, "Design of a multi-constraint PSO for resource allocation and task scheduling," *International Journal of Intelligent Systems and Applications in Engineering*, vol. 12, no. 7s, pp. 426–440, 2024.

[24] S. Zdiri, J. Chrouta, and A. Zaafouri, "An expanded heterogeneous particle swarm optimization based on adaptive inertia weight," *Mathematical Problems in Engineering*, pp. 1–24, Oct. 2021, doi: 10.1155/2021/4194263.

[25] S. Choudhary, S. Sugumaran, A. Belazi, and A. A. A. El-Latif, "Linearly decreasing inertia weight PSO and improved weight factor-based clustering algorithm for wireless sensor networks," *Journal of Ambient Intelligence and Humanized Computing*, vol. 14, no. 6, pp. 6661–6679, Jun. 2023, doi: 10.1007/s12652-021-03534-w.

[26] E. C. Pyone, S. Tangaramvong, T. H. Van, L. V. H. Bui, and W. Gao, "Comprehensive learning phasor particle swarm optimization of structures under limited natural frequency conditions," *Acta Mechanica Sinica/Lixue Xuebao*, vol. 39, no. 4, Apr. 2023, doi: 10.1007/s10409-023-22386-x.

BIOGRAPHIES OF AUTHORS



Nurul Anisa Sri Winarsih     received the S.Kom. degree in informatic engineering from Universitas Dian Nuswantoro, Indonesia, in 2016 and the M.CS degrees in computer science from Universiti Teknikal Malaysia Melaka, Malaysia, in 2017. Currently, she is a Lecturer at the Faculty of Computer Science, Universitas Dian Nuswantoro, Indonesia. Her research interests include machine learning, data mining, text mining, website developing, and artificial intelligence. She can be contacted at email: nurulanisaw@dsn.dinus.ac.id.



Ricardus Anggi Pramunendar completed studies at Dian Nuswantoro University in 2009, followed by Technical University of Malaysia Melaka in 2011, and most recently Gadjah Mada University in 2021. Skilled in computer vision, machine learning, artificial intelligence, soft computing, and image processing. Currently, he is a Lecturer at the Faculty of Computer Science, Universitas Dian Nuswantoro, Indonesia. Additionally, he is a member of IEEE, ACM, IAENG, and serves as a reviewer for SCOPUS-indexed journals. He can be contacted at email: ricardus.anggi@dsn.dinus.ac.id.



Guruh Fajar Shidik is an Associate Professor at Faculty of Computer Science, Universitas Dian Nuswantoro, Indonesia. He received the Informatic Engineering degree from Universitas Dian Nuswantoro, Indonesia in 2009. He received the master degree in computer science from Universiti Teknikal Malaysia Melaka, Malaysia in 2011. He completed his Dr. degree in computer science from the Universitas Gadjah Mada, Indonesia, in 2016. His research interests include artificial intelligence, distributed system, image processing, internet of things, big data, computation time, big data analytics, smart society, security data, machine learning, and data mining. He can be contacted at email: guruh.fajar@research.dinus.ac.id.



Budi Widjajanto received the S.P. degree from Universitas Gadjah Mada, Indonesia, in 2000 and the M.Sc. degrees from Universitas Gadjah Mada, Indonesia, in 2008. Currently, she is a Lecturer at the Faculty of Computer Science, Universitas Dian Nuswantoro, Indonesia. His research interests include machine learning, text mining, software engineering, and database desain. He can be contacted at email: budi.widjajanto@dsn.dinus.ac.id.



Muhammad Syaifur Rohman received the S.Kom. degree in informatic engineering from Universitas Dian Nuswantoro, Indonesia, in 2016 and the M.Cs. degrees in computer science from Universiti Teknikal Malaysia Melaka, Malaysia, in 2017. Currently, he is a Lecturer at the Faculty of Computer Science, Universitas Dian Nuswantoro, Indonesia. His research interests include machine learning, software engineering, website developing, mobile developing, and artificial intelligence. He can be contacted at email: syaifur@dsn.dinus.ac.id.



Danny Oka Ratmana received the S.Kom. and M.Kom degree in informatic engineering from Universitas Dian Nuswantoro, Indonesia. Currently, he is a Lecturer at the Faculty of Computer Science and Programmer in Data and Information Unit Universitas Dian Nuswantoro. His research interests include machine learning, artificial intelligence, software engineering, and website developing. He can be contacted at email: rdannyoka@dsn.dinus.ac.id.